REMARKS

This Amendment and the following remarks are intended to fully respond to the Final Office Action dated June 30, 2005. In that Final Office Action, claims 41-58 were examined, and all claims were rejected. Claims 41-58 are rejected under 35 U.S.C. §102(e) as being anticipated by Craig et al (hereinafter Craig), US 6,757,708. Claims 48 and 57 are rejected under 35 U.S.C. §103(a) as being obvious in view of Craig and further in view of "Web Services Description Language," Curbera et al., March 2001 (hereinafter Curbera). Claims 49 and 58 were rejected under 35 U.S.C. §103(a) as being obvious in view of Craig and further in view of "Metadata Activity Statement," February 2001 (hereinafter Metadata). Reconsideration of these rejections, as they might apply to the original and amended claims in view of these remarks, is respectfully requested.

In this Response, claims 41 and 50 have been amended, no claims have been cancelled, and no new claims have been added. Therefore, claims 41-58 remain present for examination.

Claim Rejections of Claims 41-58 under 35 U.S.C. § 102 and § 103

Claims 41-47 and 50-56 were rejected under 35 U.S.C. §102(e) as being anticipated by Craig. Claims 48 and 57 were rejected under 35 U.S.C. §103(a) as being obvious in view of Craig and further in view of "Web Services Description Language," Curbera et al., March 2001 (hereinafter Curbera). Claims 49 and 58 were rejected under 35 U.S.C. §103(a) as being obvious in view of Craig and further in view of "Metadata Activity Statement," February 2001 (hereinafter Metadata). The Applicants respectfully traverse these rejections.

Regarding Claim 41 and 50

Applicants respectfully traverse the section § 102(e) rejections. The amended claims necessarily preclude a finding of a prima facie case of anticipation because the requirements of a prima facie case simply are not met. Indeed, a prima facie case of anticipation can only be met when the prior art reference teaches or suggests all the claim limitations. See, MPEP § 2131. Craig does not describe, teach or suggest: automatically generating the data exchange schema data that specifies how to exchange data between the server and the client process for the data processing object generated when the source code file is compiled to generate the data processing object that provides the requested processing service.

This aspect of the invention is described in detail in the application with reference to FIG. 9 and from page 21, line 19 through page 23, line 11. Specifically, embodiments of the present invention comprise, "a textual schema description in a data exchange schema specification format . . . for specifying how schema descriptions for data being exchanged using XML are to be specified." Page 23, lines 3-6. In further embodiments, the data exchange schema, "is used to specify the XML data within the packet payload body data." Page 23, line 7. This schema data may then be transmitted to a client so that, "users may obtain these schema descriptions to determine the format and functions of the web services and corresponding input and output arguments." Page 23, lines 10-11.

The Examiner argues that Craig, on Col. 9, lines 5-40, teaches the data exchange schema. In essence, Examiner claims that the Java bean, described in Craig, is a container of object information that contains schema information. See, Office Action page 2, paragraph 5. Applicants respectfully disagree.

Craig provides two models for creating java beans. In the first model, Craig states that the bean is a "passive container for data." Col. 9, line 7. The data is data extracted from a datastore and saved into the bean by a servlet, but the data does not include object information. See, col. 9, lines 11-16. Thus, in this first model, Craig does not teach, in any way, that the bean generates or stores data exchange schema data. Examiner must be relying on the second model in Craig.

Unfortunately, Craig's second model also does not bear fruit. The Java bean in the second model contains a method to access a backend data source. See, col. 9, lines 21-22. A JSP sets the bean's input properties, executes the bean's method, and accesses the bean's output properties. However, Craig does not describe generating a data exchange schema in the bean, but simply teaches the setting or accessing of bean properties. The actual setting or accessing of tags or other bean properties is not the same as generating schema descriptions for how data being exchanged are to be specified. Examiner mistakenly correlates the process of setting the properties to the specification on how to set the properties. As an analogy, Examiner, with the same reasoning, would necessarily equate the process of driving a car to a manual describing how to drive a car. The correlation suggested by Examiner is improper.

Further, in Craig, the way the Java Beans and JSPs interact, i.e., the inputs required by each Java Bean, what services each Java Bean performs, and what output is sent back to the JSP, is previously known to the JSP. See, col. 9, lines 45-57 and specifically lines 46-53 ("The bean developer then communicates with the Web page designer as to what input properties of the bean must be set before the Web page invokes the bean's function, as well as the bean's output properties that will result from the bean executing, where the value of these output properties can then be embedded within the resulting Web page as dynamically generated content."). Thus,

Craig does not teach a data exchange schema to allow a user to determine the format and functions of the web services and corresponding input and output arguments but teaches away from it. .

Craig requires the Web page designer to ask for this information from the bean developer. The cited Col. 9 section in Craig discusses aspects of Craig that are, in fact, very different from the claimed invention. The discussion focuses on the contents of the communications between the Java Beans and the JSP. While there is some discussion of the input values and output values and their format in these sections, there is no discussion of a data exchange schema being generated by any JSP or Java Bean. Rather, the JSP is deemed to have a complete understanding of how to exchange data with the Java Beans, what input parameters to provide it from the initial request, and what output to expect in return. See, e.g., col. 9, lines 32-34 ("For each bean, the JSP first sets the bean's input properties, calls its "execute" method and then gets information by accessing the bean's output properties."). Craig simply presupposes that the JSP knows the data exchange schema of the Java Beans. In fact, Examiner admits that the function of the beans is known but then missteps and states that Craig then teaches generating a data exchange schema. See, Office Action, page 3, paragraph 5. Unfortunately, Examiner does not provide any citations to Craig to bolster this argument but seems to rely on assumptions and conjecture. Regardless, Craig does not teach generating a data exchange schema that allows a client process to determine the arguments for a web service, i.e., Craig does not generate anything that provides this information.

The Examiner should also note that any data generated by Craig occurs during the Java Bean execution. Craig does not teach or disclose generating any input value descriptions or data of any kind during compilation of the Java Bean. Claim 41 and claim 50, on the other hand, are

limited to generating the schema data, "when the source code file is compiled to generate the data processing object that provides the requested processing service." Thus, Craig does not anticipate this limitation of claim 41 and claim 50.

For the above reasons, Craig does not teach or disclose automatically generating the data exchange schema data that specifies how to exchange data between the server and the client process for the data processing object generated when the source code file is compiled to generate the data processing object that provides the requested processing service as claimed in amended claim 41 and amended claim 50. Applicants respectfully request that the Examiner withdraw this rejection and find claims 41 and its dependent claims 42-49 in a condition for allowance. Similarly, Applicants request the Examiner to allow claims 50-58 as they are computer-readable medium claims with substantially the same limitations of the method claims of claim 41-49 and are likewise not anticipated by Craig for the same reasons.

Conclusion

This Amendment fully responds to the Office Action mailed on June 30, 2005. Still, that Office Action may contain arguments and rejections and that are not directly addressed by this Amendment due to the fact that they are rendered moot in light of the preceding amendments and arguments in favor of patentability. Hence, failure of this Amendment to directly address an argument raised in the Office Action should not be taken as an indication that the Applicant believes the argument has merit. Furthermore, the claims of the present application may include other elements, not discussed in this Amendment, which are not shown, taught, or otherwise suggested by the art of record. Accordingly, the preceding arguments in favor of patentability are advanced without prejudice to other bases of patentability.

It is believed that no further fees are due with this Response. However, the Commissioner is hereby authorized to charge any deficiencies or credit any overpayment with respect to this patent application to deposit account number 13-2725.

In light of the above remarks, it is believed that the application is now in condition for allowance, and such action is respectfully requested. Should any additional issues need to be resolved, the Examiner is requested to telephone the undersigned to attempt to resolve those issues.

Respectfully submitted,

Date: August ~~30,~~ 19 2005

Tadd F. Wilson
Reg. No. 54,544
MERCHANT & GOULD P.C.
P.O. Box 2903
Minneapolis, Minnesota 55402-0903
(303) 357-1651